Estimation Methods for Ranking Recent Information

Miles Efron
Graduate School of Library & Information Science
University of Illinois, Urbana-Champaign
501 E. Daniel St, Champaign, IL 61820
01-217-265-0825
mefron@illinois.edu

Gene Golovchinsky
FX Palo Alto Laboratory, Inc.
3400 Hillview Ave, #4
Palo Alto, CA 94304
gene@fxpal.com

ABSTRACT
Temporal aspects of documents can impact relevance for certain kinds of queries. In this paper, we build on earlier work of modeling temporal information. We propose an extension to the Query Likelihood Model that incorporates query-specific information to estimate rate parameters, and we introduce a temporal factor into language model smoothing and query expansion using pseudo-relevance feedback. We evaluate these extensions using a Twitter corpus and two newspaper article collections. Results suggest that, compared to prior approaches, our models are more effective at capturing the temporal variability of relevance associated with some topics.

Categories and Subject Descriptors
H.3.3 [**Information Search and Retrieval**]: Retrieval models

General Terms
Algorithms, Experimentation

Keywords
Time, ranking algorithms, information retrieval, microblogs

1. INTRODUCTION
Recency is an important dimension of relevance for many kinds of queries. For these queries, relevant documents must not only be topically appropriate; they must also have been published in the recent past. Though research on *recency queries* is not new, these queries present an especially keen challenge in contemporary IR given the growing popularity of microblogging services such as Twitter [21]. In the context of microblog search, fielding recency queries effectively is of prime importance. This paper proposes three novel approaches to handling recency queries, examining their effectiveness both on microblog data, as well as on more traditional IR data in the form of two TREC news collections.

Previous work has shown how recency can be incorporated into IR [5] [6], particularly under the language modeling framework [15]. In this paper we build on these findings. Our contribution is a group of methods for incorporating temporal information into language modeling IR. Each of the proposed methods relies on Bayesian estimation where we use time as a factor in our retrieval model. In tests, our proposed methods work as effectively as or better than established temporal approaches for recency queries, while mitigating the risk entailed by temporally informed ranking on queries without an explicit recency bias.

SIGIR'11, July 24–28, 2011, Beijing, China.

2. PREVIOUS METHODS FOR INCORPORATING TIME INTO LANGUAGE MODELING IR
This paper's treatment of temporal factors in IR is based on the language modeling approach to document retrieval [18,26]. In particular, we rely on the query likelihood model. Given a query q and a document d we derive a score for d against q that is proportional to the probability that the (multinomial) language model that generated d also generated q:

$$Pr(d|q) \propto Pr(q|d)Pr(d) \quad \textbf{(1}$$

If we momentarily assume that the prior probability distribution is over documents is uniform, we may rank documents in decreasing order of the query's likelihood of generation by the model that generated d. Assuming that this model is a multinomial over words in the indexing language, we have the ranking function:

$$\log Pr(q|d) = \sum_{w \in q} \log Pr(w|d) \quad \textbf{(2}$$

With respect to estimating Pr($w|d$), the application of a smoothing operation has been shown to play a crucial role in successful language modeling IR. Numerous smoothing methods exist [25]. In order to minimize the influence of confounding variables in the following study, we choose the simplest smoothing method: so-called Jelinek-Mercer smoothing, which gives:

$$\hat{P}r(w|d) = (1-\lambda)\hat{P}r_{ML}(w|d) + \lambda \hat{P}r(w|C) \quad \textbf{(3}$$

where $\hat{P}r_{ML}(w|d)$ is the maximum likelihood estimator, $n(w,d)/n(d)$[1], $\hat{P}r(w|C)$ is the estimated probability of seeing word w in the collection, and λ is a tuning parameter on [0,1].

Using Eq. 2 with estimates from Eq. 3 for *ad hoc* retrieval has shown state-of-the art effectiveness, while easily admitting alterations to the retrieval process.

For instance, in their work on recency queries Li and Croft proposed using the publication date of news articles to inform a prior distribution over documents [15]. Rather than taking $Pr(d)$ in Eq. 1 to be uniform, Li and Croft propose using an exponential distribution:

$$Pr(d|t_d) = r \cdot e^{-r \cdot t_d} \quad \textbf{(4}$$

where t_d is the time in months that has elapsed since document d was published, and r is the rate parameter of the exponential distribution. The intuition of this approach is that newer documents have a higher probability (perhaps of being read) than older documents do. By using Eq. 4 as the document prior in Eq. 1

[1] In our notation, $n(w,d)$ refers to the number of times word w occurs in document d; $n(d)$ is the number of tokens in d.

Li and Croft show significant improvement in retrieval for TREC topics that are explicitly concerned with recent events.

In more recent work, Dakka, Gravano and Ipeirotis augment the standard query likelihood framework to account for time [5]. They approach the problem by considering two types of features for a given document. First, they consider w_d which consists of the lexical terms in document *d*. Second, they posit t_d which is the timestamp for *d*. With these definitions in place, we may decompose the likelihood function:

$$Pr(d|q) = Pr(w_d, t_d|q) = Pr(t_d|w_d, q) \cdot Pr(w_d|q) \quad (5$$

Making the simplifying assumption that the temporal relevance of *d* does not depend on the document's content, *w*, we drop *w* from the joint probability in Eq. 5, giving:

$$Pr(d|q) \propto Pr(w_d|q) \cdot Pr(t_w|q)$$
$$\propto Pr(q|w_d) \cdot Pr(w_d) \cdot Pr(t_d|q) \quad (6$$

which is identical to the standard query likelihood model, but with the addition of the probability of observing a time t_d given the query *q*. Eq. 6 gives a flexible way to add temporal information to document ranking.

3. OVERVIEW: THREE APPROACHES TO TEMPORAL RE-RANKING

In this paper we propose three overarching approaches to incorporating time into retrieval, paying special attention to the matter of recency queries. We offer the following approaches:

1. Query-specific exponential re-ranking
2. Temporally informed smoothing
3. Temporally biased pseudo-relevance feedback

Approach 1. The chief difference between Eq. 6 and the approach offered by Li and Croft is that Eq. 6 allows us to consider not only when a document was published, but also the relationship between that publication time and the query at hand. Under Approach 1 we apply a more aggressive temporal factor if we have evidence that our query is indeed recency-bound. We accomplish this by incorporating query-specific information to estimate the exponential rate parameter r_q to obtain the maximum a posteriori estimate, then applying this result in Eq. 6.

Approach 2. Effectively estimating language models for retrieval requires smoothing, which we accomplish using Eq. 3. This requires choosing a smoothing parameter λ. Typically this value is constant across all documents. However, in the context of recency queries it is plausible that we should smooth older documents' models more aggressively than newer documents. We elaborate on this intuition below. But the thrust of this approach is that each document's model is smoothed more aggressively for documents that are further from the target time associated with the query.

Approach 3. The relevance model framework proposed by Lavrenko and Croft [14] provides a mechanism for using (pseudo) relevance feedback in language modeling IR. A relevance model is typically a multinomial distribution over words giving $Pr(w|R)$. We usually estimate this model by an analysis of the top *k* documents returned during an initial retrieval. In this setting, $Pr(w|R)$ is estimated by a weighted average of $Pr(w|d)$ in each of the *k* returned documents. We propose using estimates of $Pr(w|d)$ based on Approach 2 to construct temporally conditioned relevance models.

4. PRELIMINARIES AND CONTEXT

To situate our approaches, we begin with several definitions.

4.1 Representing Time

First we define t_d, the time associated with a document *d*. Further, we define t^* as the most recent timestamp contained in our corpus *C*. Alternatively, we can define t^* as *now*. Thus we have:

$$t_d = \left(t^* - TIMESTAMP(d)\right) \cdot S \quad (7$$

Here *S* is a time scale appropriate to the collection, and t_d is the time that has elapsed since *d* was created. We also use the notation t_0 to refer to the earliest time observed in a corpus. For TREC data, we measure time in fractions of a month; for Twitter data we use fractions of a day.

4.2 Experimental Data and Setup

We evaluated our algorithms using three test collections: a Twitter dataset, TREC AP (disks 1 and 2), and TREC LA/FT (disks 4 and 5) using the lemur toolkit (http://lemurproject.org). We used no stemming or stop-lists (except in one case indicated below).

4.2.1 Twitter Data

We compiled corpus of tweets obtained via the Twitter API. Summary statistics of this corpus appear in Table 1.

Table 1. Summary statistics of Twitter data set.

Number of Tweets	7,168,842		
Number of Tracked Users	9,274		
Earliest Date	18 Jul 2010		
Latest Date	4 Dec 2010		
Queries		Train	Test
	Recency	21	49
	Non-temp.	31	50

The corpus was constructed by choosing an initial group of Twitter users who are, broadly speaking, interested in information retrieval, HCI, data mining, and information science. Full details of this group appear in [20]. The pool of "seed users" consisted of 48 people who posted tweets containing any of eight manually identified keywords during the week surrounding March 24, 2010 (the date of paper notifications for SIGIR 2010). Having identified these "seed" users, we expanded the pool by finding people whom these users follow on Twitter (as of June 2010). The median number of people followed by our seed users was 264; the total was 9,274. Thus the corpus contains a core of "IR community" members, broadened with the extra degree of separation. Having defined our population, we captured all tweets written by this group between July 18 and December 4.

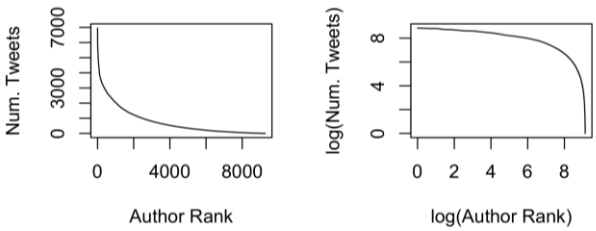

Figure 1. Distribution of Author Tweet Frequency

Figure 1 shows the number of tweets written by each author in our corpus. Somewhat surprisingly, the data do not show as pronounced a power-law distribution as we might expect, suggesting that in fact the contents of the collection was created by a diverse group.

Our retrieval experiments on the Twitter collection were based on a set of test queries which were obtained by asking two users of an experimental Twitter search engine to create queries of two types:

1. Recency Queries: Queries where relevant tweets were necessarily written very recently. These queries were created on December 4, 2010.
2. Non-temporal Queries: Queries where relevant tweets need not be new[2].

Each query author created 25 recency queries and 25 non-temporal queries.

Finally, for each query we obtained relevance judgments via the Amazon Mechanical Turk (AMT) service. We created judgment pools by retrieving n=50 documents using four retrieval models: query likelihood, query likelihood with exponential recency priors, BM25 [19], and KL divergence with pseudo-RF using the relevance model approach [14,24]. These models were chosen in efforts to achieve suitable diversity in the judging pools.

For each judgment we recorded basic demographic information of the judge, a numeric relevance score measured on a three-point Likert scale: 0 (not relevant), 1 (maybe relevant), 2 (relevant), with an option for *I don't know.* Each query/document pair was judged by six workers; queries that received at least one *I don't know* response were omitted. Final numeric relevance scores were obtained by two methods:

- **Averaging:** We counted as relevant documents with an average of at least 1.5 (an admittedly heuristic value).
- **Voting:** The final score was found by a majority vote. However, for a judgment to receive a non-zero score, at least three of the judges needed to agree on its value and none could have given a zero score. Documents that scored 2 were counted as relevant.

For quality control, only AMT workers with 97% or greater approval rates and completion of at least 50 tasks were allowed to participate. Additionally, we omitted judgments that were completed in less time than the first quartile among our sample (seven seconds). This led us to omit 7,843 judgments, leaving a total of 31,454 (also omitting *I don't know* responses).

All judgments for recency queries were gathered on December 5, 2010 so that workers would have suitable context for making their decisions (they were invited to consult the current Twitter stream while making their relevance assessments).

Prior to building our test queries and relevance judgments, we compiled a set of training queries and judgments using the same approach described above, except that the authors themselves created the training queries, and these were created and judged on November 3-4, 2010.

[2] In fact, several of our query authors' non-temporal queries did end up showing some temporal bias. In some cases, this was a slight "recency" concern. In other cases, relevant documents tended to cluster around a particular, past time period.

4.2.2 TREC Data

Because recency queries form an important part of many retrieval settings we also tested our approaches on two sets of news text gathered for TREC [7]. Details of the TREC collections are given in Table 2. We used topic titles as the query text.

Table 2. Summary statistics of TREC data sets.

	AP	LA/FT
Documents	AP (disks 1 & 2)	LA Times and Financial Times (disks 4 & 5)
Doc. Count	164,597	342,054
Dates	1 Jan '88 – 31 Dec '89	23 Apr '91 – 31 Dec '94
Topics	101-200	351-450 (test), 301-350 (train)
Recency queries	20 (test only)	24 (test), 17 (train)
Non-temp. queries	72 (test only)	65 (test), 30 (train)

We classified each topic as "recency" or "non-recency" based on the temporal distribution of each query's relevant documents. If at least 2/3 of the relevant documents appear prior to the median document time, the query was considered a candidate for recency status. This was an admittedly *ad hoc* threshold, but it was chosen both for plausibility and in order to generate a suitable number of candidates. We then manually examined each query to determine if it had a *bona fide* temporal dimension to its relevance. Only queries that met the 2/3 criterion and seemed temporally bound were classified as recency queries. [3] All others were called "non-temporal" although they may indeed have temporal qualities aside from recency. Finally, we only retained queries with at least 10 relevant documents. This final number of selected queries for each TREC collection is shown in Table 2. We used TREC topics 301-350 as training data to estimate parameters. The procedure described above yielded 17 recency queries and 33 non-temporal queries, which were evaluated on the LA.FT data.

4.2.3 Baseline Systems

In Section 7 we analyze the effectiveness of our proposed methods of handling recency queries. To contextualize our results, we report comparisons against two baseline systems. The first approach is the simple query likelihood model (QL) of Eq. 2. We also report results obtained by the application of time-based exponential priors (EXP) as outlined by Li and Croft [15]. In our discussion of relevance models, we replace the QL baseline with the Kullback-Leibler divergence (KLD) model [24].

5. TIME AND RETRIEVAL

5.1 Maximum *a posteriori* estimation for exponential re-ranking

In a context such as Twitter, where relevance often has a recency component, rewarding newer documents during retrieval has

[3] Recency queries: AP (104, 116, 117, 122, 132, 133, 137, 139, 140, 148, 154, 164, 174, 175, 188, 192, 195, 196, 199, 200); FT/LA, train (06, 307, 311, 316, 319, 320, 321, 324, 326, 329, 331, 334, 337, 339, 340, 345, 346); FT/LA, test (351, 352, 357, 373, 376, 378, 387, 389, 391, 401, 404, 409, 410, 414, 416, 421, 428, 434, 437, 443, 445, 446, 449, 450)

obvious appeal. On the other hand, a time-only ranking as used by Twitter search fails to capture differences in tweets' relevance to the query. Using an exponential distribution to accomplish a blending of time and language model (Eq. 3) has been shown to be effective in previous research. However, the typical approach to using the exponential distribution as a document prior by definition ignores query-specific concerns. We hypothesize that the aggressiveness of the exponential penalty should hinge on the extent to which a particular query is sensitive to recency.

Assume that we are using Eq. 6 as our document ranking function. We can combine the approaches of Li and Croft and Dakka et al. by letting $Pr(t_d|q)$ follow an exponential distribution. Ignoring the query-independent document prior, we have the ranking function:

$$Pr(d|q) \propto Pr(q|d) \cdot r_q \cdot e^{-r_q \cdot t_d} \quad \textbf{(8}$$

with an exponential rate parameter r_q. Note that the formalization of Eq. 6 allows us to bring in a query-specific rate parameter. This is important because it has been shown that the performance of an exponential-based age penalty depends strongly on the parameterization of the distribution. Li and Croft found that setting r=0.01 yielded strong performance, while values even slightly larger than this degraded performance significantly.

A second reason that using a query-specific rate parameter is desirable is that a rate parameter leading to improved performance on recency queries appears (on our microblog data) to degrade performance for non-temporal queries. As an example, we tuned the rate parameter for the exponential prior method to *r=0.015* to optimize performance of recency queries on MAP. This parameterization led to the results in Table 3. All comparisons between QL and EXP in are significant at p<0.01 using a randomization test.

These results show two things: For recency queries, applying an exponential prior improves retrieval on our training recency queries, whereas for non-temporal queries, the exponential prior leads to a stark decrease in performance.

Table 3. Retrieval effectiveness using query likelihood and exponential priors. Relevance based on averaging (Sec. 4.2)

	Recency Queries			Non-Temporal Queries		
	MAP	**Rprec**	**NDCG**	**MAP**	**Rprec**	**NDCG**
QL	0.321	0.339	0.570	0.424	0.529	0.600
EXP	0.354+	0.373+	0.597+	0.337-	0.439-	0.519-

The decline in performance for non-temporal queries is not surprising, as the temporal re-ranking dilutes the influence of textual similarity in favor of a temporal factor that is presumably not important for these queries.

To compensate, we turn to Eq. 8 to derive an exponential distribution that penalizes older documents to an extent that depends on the strength of evidence that we are dealing with a recency query. We do this in a two-pass approach. That is, we retrieve k documents using QL, estimate r_q based on these, and then re-rank the k documents according to Eq. 8.

For a query q let $T = \{t_1, t_2, \cdots, t_k\}$ be the times associated with the top k documents returned using simple QL (throughout this paper we set *k=500*). With these times in hand we have the maximum likelihood estimator for the exponential rate parameter:

$$\hat{r}_q^{ML} = \frac{1}{\bar{T}} \quad \textbf{(9}$$

where $\bar{T}$ is the sample mean of T.

For Approach 1, our starting point is the hypothesis that query-specific temporal information can help us arrive at an estimate of the rate parameter that is more appropriate than a one-size-fits all approach. To illustrate this point, we analyzed the MLEs obtained from the recency and non-temporal training queries in our microblog data. We found $\hat{r}_{recency}^{ML} = 0.029$ and that $\hat{r}_{non-temp}^{ML} = 0.017$. The p-value on a one-sided t-test between the MLE rate parameter estimates for recency and non-recency queries was 0.057. In other words, the top documents retrieved for recency queries tended to be newer than documents retrieved for non-temporal queries. This in turn would cause us to penalize older documents more strongly for recency queries, as compared to the penalty applied for non-temporal queries.

In practice, however, we found that the MLEs obtained from our query information led to models that lent too much influence to time, eclipsing lexical query similarity (e.g. 0.029 leads to an overwhelming age penalty). To mitigate this problem, we propose a Bayesian approach, replacing the MLE with the maximum *a posteriori* estimate of r_q.

We note that the conjugate prior of the exponential is the gamma distribution with parameters α and β (our hyperparameters) yields an estimate for the exponential rate parameter [3]:

$$r_q^{MAP} = \frac{\rho + k - 1}{\sigma + \sum_{i=1}^{k} t_i} \quad \textbf{(10}$$

where ρ and σ are the rate and shape parameters of the geometric distribution, respectively. With this formalization in place we may use r_q^{MAP} in Eq. 8. The summation over the observed document times in the denominator of Eq. 10 should reduce the temporal influence for non-temporally bound queries. We refer to this approach as Bayesian Exponential Ranking (BEX).

5.2 Temporally-smoothed language models

In Section 2 we described the process of smoothing language models for document ranking (Eq. 3). The Jelinek-Mercer approach to smoothing is governed by a smoothing parameter λ that guides the amount of influence that the "background" probability $Pr(w|C)$ plays in the estimation of $Pr(w|d)$. Our second approach to incorporating time into retrieval for recency queries is to smooth older documents' language models more aggressively than newer documents' models. The rationale for this approach is that as documents age, we have less confidence in their precise lexical content. After many months, perhaps an author would use different words to describe the same topic.

We may consider the mixing parameter λ in Eq. 3 as the probability of choosing $Pr(w|C)$. That is, the author chooses the background model with probability λ, otherwise favoring the MLE. Thus the choice of the MLE or the collection language model is (under this interpretation) guided by a binomial distribution with probability parameter λ. Typically we treat this mixing parameter as a heuristic tuning quantity. But we may instead approach it as an estimation problem. With respect to the subject of this paper, we can replace Eq. 3 with:

$$\hat{P}r(w|d) = (1-\lambda_t)\hat{P}r_{ML}(w|d) + \lambda_t \hat{P}r(w|C) \quad \textbf{(11}$$

where we have replaced λ in Eq. 3 with λ_t a quantity that depends on the time associated with document *d*. The older *d* is (i.e. the larger t_d), the larger λ_t.

To guide this estimation we offer the following scenario. Imagine that at time t^* we want to re-express the content of document *d*. To do so, we generate text from its language model. However, we consider documents in our collection to be of two types: *old* and *new*. If document *d* is "old," we generate a word *w* according to $Pr(w|C)$. If the document is "new" we trust the observed word counts and generate text according to the MLE.

Given this scenario we may understand λ_t as the parameter of a binomial distribution, where a "success" corresponds to finding that *d* is old. To estimate λ_t we have the maximum likelihood estimator:

$$\hat{\lambda}_t^{MLE} = \frac{n(t<t_d)}{n(D)} \quad \textbf{(12}$$

That is, the estimate is the number of documents that are newer than *d* divided by the total number of documents in the collection.

Earlier, however, we took a Bayesian approach to parameter estimation, and we propose doing so again here. Specifically, we assume that we have a prior belief about λ. In the standard language modeling approach we set λ to some constant (in this paper we used λ=0.4). Thus we could say that we believe that λ is likely to be in the vicinity of 0.4. To formalize this intuition, we assume that λ follows a beta distribution (because beta is the conjugate prior of the binomial) with parameters α and β. Again referring the reader to [3] for the derivation, we have the maximum *a posteriori* estimator:

$$\hat{\lambda}_t^{MAP} = \frac{n(t<t_d)+\alpha-1}{n(D)+\beta-\alpha-2} \quad \textbf{(13}$$

For our temporal smoothing query likelihood approach (TSQL), we use the maximum *a postiori* estimate of λ_t of Eq. 11 to estimate $Pr(w|d)$ in the query likelihood model of Eq. 2.

5.3 Temporally conditioned relevance models

A common way to use relevance feedback in language modeling IR relies on Lavrenko and Croft's relevance model formalization [14]. If our query contains *n* tokens $q_1 \ldots q_n$ the probability of a word *w* under a relevance model is $Pr(w, q_1 \ldots q_n)$. Estimating this conditional probability requires us first to estimate the joint probability $Pr(w, q_1, \ldots, q_n)$. For purposes of relevance feedback, the estimation is typically carried out only over the top *k* documents found during an initial retrieval, giving:

$$Pr(w, q_1 \ldots q_n) = \sum_{i=1}^{k} Pr(d_i) Pr(w|d_i) \prod_{j=1}^{n} Pr(q_j|d_i) \quad \textbf{(14}$$

This allows us to define the probability of word *w* under the relevance model *R* as:

$$Pr(w|R) = \frac{Pr(w, q_1 \ldots q_n)}{Pr(q_1 \ldots q_n)} \quad \textbf{(15}$$

From a practical standpoint, relevance feedback typically proceeds by limiting $Pr(w|R)$ to contain non-zero probabilities only for the *f* feedback terms that have the highest values in Eq. 15. Then a second round of retrieval is conducted using the induced relevance model. Typically this is done by ranking documents in increasing order of the Kullback-Leibler divergence between their language models and the relevance model.

Li and Croft proposed integrating recency into relevance models through the quantity $Pr(d_i)$ in Eq. 14. Following their use of document priors, Li and Croft use the exponential distribution for $Pr(d_i)$ using Eq. 4. We refer to this approach as exponential relevance models (EXRM) in contrast to a baseline, non-temporal relevance modeling approach (RM).

However, we propose an alternate approach to incorporating time into relevance models. In many applications, the relevance model of Eq. 15 is interpolated with the original query model before the second retrieval via:

$$\hat{P}r(w|R) = (1-\mu)Pr(w|q) + \mu Pr(w|R) \quad \textbf{(16}$$

where μ is a mixing parameter that plays a role similar to λ in Eq. 3. A large value of μ gives heavy weight to the feedback terms, while a small μ is more conservative, retaining the influence of the original query. (For the RM and EXRM approaches, use Eq. 16 to estimate word probabilities under the relevance model). This is the relevance feedback approach used by the Indri search engine, for instance. Of course, in Eq. 16 μ is another tunable parameter. As in our temporally-smoothed document language models, we propose letting the timestamps of retrieved documents guide the smoothing process, giving more weight to feedback terms when the feedback query retrieves newer documents.

Our approach to setting μ is as follows. Assume we have an initial query *q*. We build a relevance model based on *q* using Eq. 15, performing a second retrieval using the KL divergence method with this relevance model. This yields a vector of feedback document times, $\dot{t}_{fb} = \{t_{fb1}, \cdots, t_{fbk}\}$ where *k* is the number of feedback documents.

In our estimation of μ we are guided by motivation similar to Eq. 12's, asking, for each element of $\dot{t}_{fb}$, how many documents are older? As before, we consider μ in Eq. 16 as a binomial parameter. To estimate this parameter, we begin with maximum likelihood, where success is the observation that a particular feedback document f_{fbi} is newer than a particular document from the collection at large. We thus have the estimate

$$\hat{\mu}^{MLE} = \frac{\sum_{i=1}^{k} n(t>t_{fbi})}{k \cdot n(d)} \quad \textbf{(17}$$

where the sum in the numerator of Eq. 17 is taken over all *k* feedback documents For simplicity we refer to the numerator of Eq. 16 as $S(\dot{t}_{fb})$. This allows us to define the maximum a posteriori estimate of μ:

$$\hat{\mu}^{MAP} = \frac{S(\dot{t}_{fb})+\alpha-1}{k \cdot n(D)+\beta-\alpha-2} \quad \textbf{(18}$$

with hyperparameters α and β. Thus we define our temporally smoothed relevance model as:

$$\hat{P}r(w|R) = (1-\hat{\mu}^{MAP})Pr(w|q) + \hat{\mu}^{MAP} Pr(w|R) \quad \textbf{(19}$$

which we call TSRM.

6. PARAMETERIZATIONS

Using the training data described in Section 3, we tuned the parameters that guide the models that we have proposed. The outcome of this process appears in Table 4. Because optimal values were very close for the TREC and Twitter data, we have chosen a single parameterization for both corpora.

Table 4. Model Parameters. Values chosen or derived for best performance (MAP) on training data.

Param	Description	Eqs.	Value
n	Num. Docs retrieved per query	NA	100
k	Num. top docs. Used for calculations	9, 13	20
r	Rate parameter for exponential priors	4	0.015 (TREC), 0.01 (Twitter)
λ	Parameter for Jelinek-Mercer smoothing	3	0.4
ρ	Effective sample size for exponential MAP estimate (BEX)	9	100
σ	Shape parameter for exponential MAP estimate (BEX)	9	Derived
β	Effective sample size for binomial MAP estimate (TSQL, TSRM)	10, 17	2**n(D)* (TSQL) 250 (TSRM)
α	Hyperparameter for binomial MAP estimate (TSQL, TSRM)	10, 17	derived2
μ	Mixing parameter for relevance models (RM, EXRM)	15	0.4
f	Number of expansion terms used in pseudo-RF	NA	10

Throughout this paper we set λ=0.4 in Eq. 3 (for standard smoothing). For time-specific smoothing we choose α and β such that $(\alpha - 1)/(\beta - \alpha - 2) = 0.4$. This makes the time-smoothed models directly comparable to the standard language models smoothed with λ=0.4, leaving only the magnitude of β as a tunable parameter. For simplicity, we take the same approach for estimating μ in Eq. 18. For our query-specific exponential re-ranking, we choose hyperparameters such that $(\rho - 1)/\sigma = 0.015$ for the TREC data and 0.01 for the Twitter data.

To avoid including very common words in expanded queries, we applied a standard stoplist when constructing relevance models. For the TREC data the stopwords consisted of the standard list included with the lemur toolkit. For the Twitter data we supplemented this list with a brief customized stoplist that removed common hostnames, file extensions, etc.

7. EXPERIMENTAL RESULTS

To assess the effectiveness of the proposed temporal retrieval models, we ran experiments using the data sets described in Section 4. Notation and abbreviations that appear in our data reporting are given in Table 5. In the following tables, results of non-temporal baseline runs are shown in grey. Cells marked with + (–) indicate a statistically-significant ($p<0.05$) improvement (decline) versus the related baseline using a randomization test. It is worth noting that in some cases, the score (e.g. MAP) of a run *a* whose improvement over the baseline is larger than run *b*'s does not see statistical significance, while run *b* does. This is due to high variance over the queries for run *a*.

We report three effectiveness metrics, mean average precision (MAP), R-precision (Rprec), and normalized discounted cumulative gain (NDCG) to show the effect of different models with respect to both recall- and precision-based considerations.

The two highlighted rows of Table 5 show our baseline systems (QL and RM). Thus the effectiveness of EXP, BEX, and TSQL is gauged against QL and EXRM and TSQL are compared to RM. We refer to the QL and associated runs as *term-based*, referring to the runs based on relevance models as *feedback runs.*

Table 5. Abbreviations of Experimental Systems. Baseline systems are highlighted in grey.

Abbreviation	**Description**
QL	Query likelihood (non-temporal)
EXP	Exponential priors as in [15]
BEX	Bayesian exponential ranking
TSQL	Time-smoothed query likelihood
RM	Relevance models (non-temporal)
EXRM	Exponential relevance models as in [15]
TSQL	Time-smoothed relevance models

The starkest result in Table 6 through Table 8 is the difference in performance between recency and non-temporal queries. For recency queries, all three non-feedback models improved MAP for all datasets significantly. Results for R-precision and NDCG were also promising for these runs with all models scoring higher than QL, though only occasionally at statistically significant levels. Moreover, all three models were comparable on recency queries. The difference in effectiveness (on any of our three measures) between EXP and BEX or TSQL was never observed to be statistically significant on the recency queries.

Table 6. Retrieval Effectiveness on TREC AP Data.

	Recency Queries			**Non-Temporal Queries**		
	MAP	**Rprec**	**NDCG**	**MAP**	**Rprec**	**NDCG**
QL	0.198	0.259	0.401	0.150	0.213	0.302
EXP	0.204	0.265	0.408	0.145-	0.208	0.296-
BEX	0.205+	0.265	0.408+	0.147	0.209	0.300
TSQL	0.203+	0.263	0.405	0.148	0.210	0.302
RM	0.267	0.318	0.469	0.192	0.262	0.357
EXRM	0.266	0.317	0.463	0.191	0.262	0.357
TSRM	0.272+	0.323+	0.473+	0.192	0.263+	0.358

However, the picture is quite different when we turn to the non-temporal queries. The method based on exponential document priors saw a statistically significant decline in MAP in comparison with the non-temporal QL for all tested non-temporal queries. Results for R-precision and NDCG are also lower for EXP than for QL in all cases.

Turning to the models proposed in this paper, the BEX approach alleviated the risk of temporal conditioning of search results for in comparison to EXP. As we hypothesized, the rate parameter of the exponential in Eq. 8 was moderated for non-temporal queries, leading to a diminished impact of time in these runs. BEX *did* show declines in performance for recency queries compared to QL. But these declines were less severe than in the case of EXP. Whereas all EXP declines in MAP were statistically significant, only those on the Twitter data were significant for BEX.

The most encouraging result comes from our temporally smoothed query likelihood model (TSQL). TSQL saw a statistically significant decline in performance only once in our experiments (with respect to R-precision on the LA/FT data. In all other cases, the decline in performance incurred by applying temporal smoothing to non-temporal queries was negligible.

The bottom three rows of Table 6 show that relevance feedback hindered retrieval on the Twitter data, with MAP significantly lower for RM than for QL in all cases but one (non-temporal by voting). This poor performance may be due to short documents in the Twitter collection. Thus with respect to relevance feedback models, we focus on the results from Table 7 and Table 8. As Li and Croft found, the EXRM approach shows little improvement over the baseline RM. Several runs, however, show substantial improvement with TSRM. As expected, the bulk of this improvement occurs on recency queries; temporal smoothing appears to have negligible impact on non-temporal queries.

8. DISCUSSION

Two main results emerge from the data reported above: BEX and TSQL improve recency query performance to nearly the same extent as EXP, and BEX and (especially) TSQL are more robust against failure than EXP when applied to non-temporal queries.

Under BEX, bringing query-specific information to specifying the exponential parameter allows recency information to privilege newer documents while tempering this influence for non-temporal queries. Comparing EXP to BEX in Tables 6 and 7 speaks to this.

Table 8. Retrieval Effectiveness on TREC LA/FT Data.

	Recency Queries			Non-Temporal Queries		
	MAP	**Rprec**	**NDCG**	**MAP**	**Rprec**	**NDCG**
QL	0.175	0.225	0.324	0.142	0.208	0.306
EXP	0.187+	0.238	0.332	0.134-	0.198-	0.209-
BEX	0.186+	0.241+	0.331+	0.138	0.201-	0.297-
TSQL	0.184+	0.241+	0.326	0.138	0.199-	0.300
RM	0.193	0.244	0.335	0.152	0.222	0.319
EXRM	0.197	0.248	0.340	0.151	0.219-	0.318
TSRM	0.196+	0.244	0.339+	0.152	0.219-	0.318

However, TSQL showed still more robustness and promise. While prior-based methods (EXP and EXRM) gave significant improvements over their baselines four times, TSQL saw 12 significant improvements. Also, EXP and EXRM reduced effectiveness significantly in eleven runs, versus 2 for TSQL (no significant declines for TSQL on MAP). While applying temporal priors to relevance models yielded no discernable improvement over baseline RM, TSRM was able to improve over the baseline RM runs on the TREC data.

8.1 Sensitivity of Models to Parameterizations

Table 4 reminds us that we have defined and set many parameters in the course of this paper. An obvious question is, how sensitive are the proposed methods to the parameterization of their models? Figure 2 shows the effect of changing the parameterization for each of the approaches we have outlined (except relevance feedback). The plotted data are for the recency training queries on both our TREC and Twitter corpora. In each panel, the blue line shows mean average precision as we change the rate parameter for Li and Croft's exponential priors (EXP). The blue solid line shows MAP for the Bayesian estimates of time-sensitive exponentials (BEX). The dashed red line shows the sensitivity of the temporal smoothing method (TSQL). Finally, the horizontal black dotted line is MAP observed for standard query likelihood.

Of special interest in Figure 3 is the behavior of the temporal smoothing model. We can see that while the two methods based on the exponential distribution are highly sensitive to their

Table 7. Retrieval Effectiveness on Twitter Data. Relevance judgments based on averaging and voting methods (Section 4.2)

	Relevance by Averaging						Relevance by Voting					
	Recency Queries			Non-Temporal Queries			Recency Queries			Non-Temporal Queries		
	MAP	**Rprec**	**NDCG**	**MAP**	**Rprec**	**NDCG**	**MAP**	**Rprec**	**NDCG**	**MAP**	**Rprec**	**NDCG**
QL	0.340	0.409	0.576	0.336	0.358	0.535	0.325	0.390	0.573	0.276	0.285	0.488
EXP	0.364+	0.420	0.596	0.305-	0.338	0.486-	0.343+	0.411+	0.589	0.244-	0.267-	0.430-
BEX	0.362+	0.421	0.597+	0.317-	0.349	0.499-	0.344+	0.406+	0.591	0.256-	0.274	0.444-
TSQL	0.361+	0.418	0.594	0.335	0.360	0.528	0.347+	0.408+	0.589	0.275	0.287	0.478
RM	0.313	0.399	0.549	0.332	0.367	0.517	0.312	0.383	0.541	0.279	0.298	0.477
EXRM	0.313	0.394	0.538	0.339	0.360	0.526	0.304	0.377	0.534	0.277	0.302	0.477
TSRM	0.315	0.390	0.537	0.337	0.359	0.525	0.306	0.384	0.532	0.277	0.299	0.475

parameterization, TSQL is much less so. In fact, its MAP remains above the baseline QL for all tested parameterizations. We tested parameterizations up to an effective sample size of twenty times $n(D)$. As Eq. 13 shows, so long as we define the hyperparameters such that $\alpha/(\beta-\alpha-1)=0.4$, then as we increase β (the effective sample size) the TSQL model probabilities will simply approach the λ=*0.4* of the standard QL model.

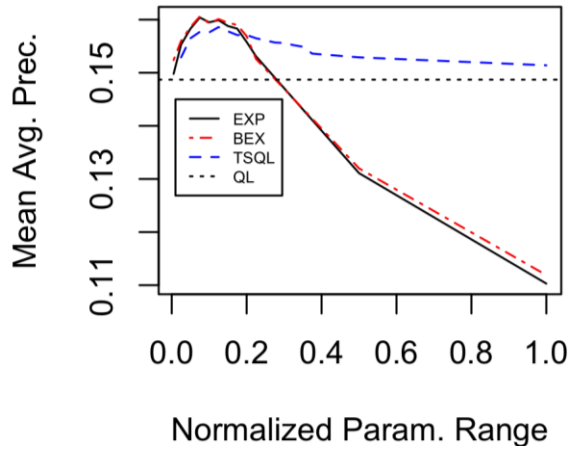

Figure 2. Effect of model parameters on MAP for TREC training data. The x-axis shows the parameters of each model normalized to appear on the same scale.

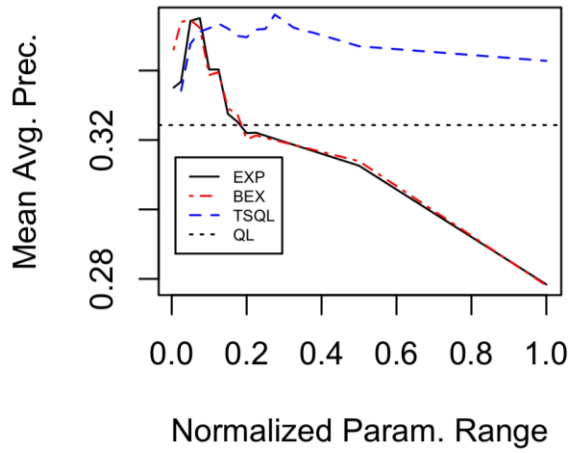

Figure 3. Effect of model parameters on MAP for Twitter training data.

Table 9. Term counts and lengths for documents in Figure 4

	petroleum	exploration	falkland	*n(d)*
FT932-16710	2	1	5	292
FT934-4629	1	2	0	79

8.2 Performance of Time-Smoothed Language Models

In Section 7 we presented evidence that temporal smoothing (TSQL) admits recency considerations into retrieval more effectively than the use of exponential priors (EXP) does. Figure 4 shows the effect of time on the estimated query likelihood, as illustrated by TREC topic 351. We selected two documents, FT932-16710 (the top-ranked document) and FT934-4629 (10th). However, instead of using each document's real timestamp, we replaced its time with the number shown on the *x*-axis of Figure 4, calculating the time-adjusted query likelihood. This allows us to gauge the change in each document's score as it gets "older."

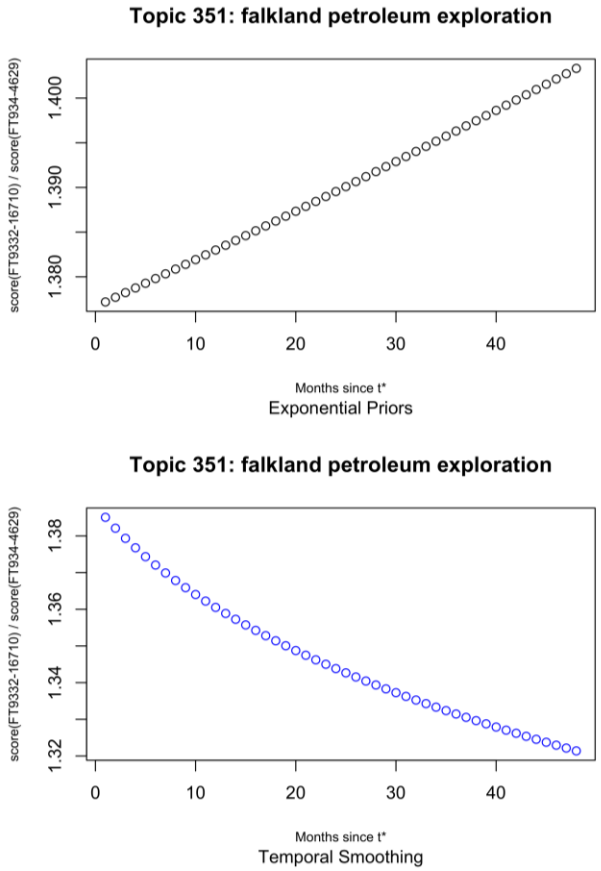

Figure 4. Effect of time on relative document scores using Exponential Priors (EXP, top) and Temporal Smoothing (TSQL, bottom). Points are the ratio of query likelihood at simulated time for documents FT932-16710 and FT934-4629.

Figure 4 shows that time affects document scores quite differently under EXP versus TSQL. As a point of reference, using a constant λ=0.4, FT932-16710 has a higher likelihood than FT934-4629; the ratio of these quantities is 1.27. Thus all of the ratios in Figure 4 are greater than 1.0. However, as we alter the timestamps on each document—making them appear older—we see that under EXP, the documents become increasingly and linearly *less* similar. On the other hand, temporal smoothing *increases* the similarity between the two documents given that they are "old."

Our interpretation is that TSQL integrates time into scoring in a more realistic way than EXP does. For documents that are "new," TSQL adds stark separation between the documents in Figure 4. But if documents with the same lexical content were older, TSQL's temporal influence diminishes.

9. RELATED WORK

In addition to the work that we have cited in the previous discussion, a great deal of interest has guided research into temporal dimensions of IR. Time, researchers have found, enters into retrieval in several ways—shaping the notion of relevance, constituting a valuable source of evidence in ranking, and helping developers identify suitable algorithms for particular settings.

Temporal factors have been shown to provide leverage in several IR-related problems, including document ranking [6] and query difficulty prediction [8].

Web retrieval has seen particular interest in temporal matters. Change over time raises the matter of page *freshness*, as described in [4]. In other work, it has been shown that changes in Web documents can bear both on how users perceive relevance [20] and how search engines model document contents [1].

A good deal of recent work has analyzed the problem of search in various social settings [13]. Problems such as collaborative and social search [9,11] draw attention to the role that relationships between people play in a variety of search settings. Other work pays specific attention to the social aspects of information seeking in microblog settings [8,16]. More traditional treatments of IR in microblogs is offered in [2,7,10] and [23].

10. FUTURE WORK

The work presented in this paper builds on prior attempts to inject time into queries. We've taken another step toward making time a first-class citizen of the retrieval space, but much remains to be done. The evaluation framework we adopted might be improved by removing some of the arbitrary cutoff parameters, for example.

In addition to recency queries, we might consider queries that prefer older documents. For certain precedence searches (e.g., patent search, trademark search, e-discovery) one might wish to know when certain concepts were first mentioned. The formalisms described here might be inverted to bias the results toward older documents. One challenge is to establish a useful reference point in the past that corresponds to the present in recency queries.

There are other ways of injecting time into search results. We can, for example, represent t_d as a separate dimension in a rank-then-combine approach described by Pickens and Golovchinsky [17]. In this approach time becomes yet another feature on which documents can be ranked. The weight assigned to this dimension can either be calculated analogously to Eq. 12, or can be controlled directly by the user in an exploratory search setting.

Temporal smoothing appears to be an effective way to add time to our consideration during document ranking. However, in this paper we have only pursued a simple application of this idea. In future work it will be of interest to apply the ideas introduced here to other smoothing methods such as Bayesian smoothing with Dirichlet priors. In this case, the temporal influence would guide retrieval in a more complex way than it does using Jelinek-Mercer. This would be especially interesting due to the different nature of the document types we have analyzed. Tweets and news articles obviously differ with respect to length, and thus word frequency. It will be interesting to analyze how temporally informed Dirichlet smoothing bears on such varied documents.

11. CONCLUSION

Though time has long played a role in IR, new problems such as microblog search change the nature of temporal retrieval. In this paper we proposed methods based on Bayesian estimation for incorporating time into language modeling IR. In one approach (BEX), we added a query-specific estimation procedure to the use of an exponential penalty for document age. We also used time to perform document-specific language model smoothing (TSQL). Finally, we used this method to induce temporally smoothed relevance models (TSRM). The methods we propose perform at state of the art effectiveness for recency queries, while showing more robustness than established methods when applied to non-temporal queries.

12. ACKNOWLEDGMENTS

This research was supported in part by a Google academic research award.

13. REFERENCES

[1] Adar, E. Teevan, J., Dumais, S.T., and Elsas, J.L., 2009. The web changes everything: understanding the dynamics of web content. *Proceedings of the Second ACM International Conference on Web Search and Data Mining* (New York, NY, USA, 2009), 282–291.

[2] Bernstein, M. Suh, B., Hong, L., Chen. J., Kairam, S., and Chi, E. 2010. Eddi: Interactive Topic-Based Browsing of Social Status Streams. *UIST 2010: ACM Symposium on User Interface Software and Technology*. (forthcoming) (2010).

[3] Bolstad, W.M. *Introduction to Bayesian Statistics*. Wiley-Interscience.

[4] Dai, N. and Davison, B.D. 2010. Freshness matters. *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '10* (Geneva, Switzerland, 2010), 114.

[5] Dakka, W., Gravano, L., Ipeirotis, P., Answering General Time-Sensitive Queries, *IEEE Transactions on Knowledge and Data Engineering*, vol. 99, no. PrePrints, 2010.

[6] Dong, A. Zhang, R., Kolari, P., Bai, J., Diaz, F., Chang, Y., Zheng, Z., and Zha, H. 2010. Time is of the essence: improving recency ranking using Twitter data. *Proceedings of the 19th international conference on World wide web* (New York, NY, USA, 2010), 331–340.

[7] Efron, M. 2010. Hashtag retrieval in a microblogging environment. *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval* (Geneva, Switzerland, 2010), 787-788.

[8] Efron, M. and Winget, M. 2010. Questions are content: A Taxonomy of Questions in a Microblogging Environment. *Proceedings of the 2010 Annual Meeting of the American Society for Information Science and Technology*. (2010).

[9] Evans, B.M. and Chi, E.H. 2008. Towards a model of understanding social search. *Proceedings of the 2008 ACM conference on Computer supported cooperative work* (San Diego, CA, USA, 2008), 485-494.

[10] Golovchinsky, G. and Efron, M. 2010. Making sense of Twitter search. *Proc. CHI2010 Workshop on Microblogging: What and How Can We Learn From It? April 11, 2010.* (2010).

[11] Horowitz, D. and Kamvar, S.D. 2010. The anatomy of a large-scale social search engine. *Proceedings of the 19th international conference on World wide web* (Raleigh, North Carolina, USA, 2010), 431-440.

[12] Jones, R. and Diaz, F. 2007. Temporal profiles of queries. *ACM Transactions on Information Systems*. 25, 3 (2007), 14-es.

[13] King, I., and Li, J. 2009. Proceeding of the 2nd ACM

workshop on Social web search and mining. (Hong Kong, China, 2009), 70.

[14] Lavrenko, V. and Croft, W.B. 2001. Relevance based language models. *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval* (New York, NY, USA, 2001), 120–127.

[15] Li, X. and Croft, W.B. 2003. Time-based language models. *Proceedings of the twelfth international conference on Information and knowledge management - CIKM '03* (New Orleans, LA, USA, 2003), 469.

[16] Morris, M.R., Teevan, J., and Panovich, Katrina. 2010. What do people ask their social networks, and why?: a survey study of status message q&a behavior. *Proceedings of the 28th international conference on Human factors in computing systems* (Atlanta, Georgia, USA, 2010), 1739-1748.

[17] Pickens, J. and Golovchinsky, G. 2008. Ranked feature fusion models for ad hoc retrieval. *Proceeding of the 17th ACM conference on Information and knowledge management* (New York, NY, USA, 2008), 893–900.

[18] Ponte, J.M. and Croft, W.B. 1998. A language modeling approach to information retrieval. *SIGIR 1998: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. (1998), 275-281.

[19] Robertson, S.E. Walker, S., Jones, S., Hancock-Beaulieu, M.M., and Gatford, M. 1994. Okapi at TREC-3. *In Proceedings of the Third Text REtrieval Conference (TREC 1994)* (1994).

[20] Teevan, J., Dumais, S.T., and Liebling, D.J. 2010. A longitudinal study of how highlighting web content change affects people's web interactions. *Proceedings of the 28th international conference on Human factors in computing systems - CHI '10* (Atlanta, Georgia, USA, 2010), 1353.

[21] Twitter. *http://twitter.com*. Accessed: 12-02-2010.

[22] Voorhees, E.M. 2007. TREC: Continuing information retrieval's tradition of experimentation. *Commun. ACM*. 50, (Nov. 2007), 51–54.

[23] Yang, J. and Leskovec, J. 2011. Temporal variation in online media. *ACM Internation Conference on Web Search and Data Mining (WSDM '11)* (2011).

[24] Zhai, C. and Lafferty, J. 2001. Model-based feedback in the language modeling approach to information retrieval. *CIKM 2001: Proceedings of the tenth international conference on Information and knowledge management*. (2001), 403-410.

[25] Zhai, C. and Lafferty, J. 2004. A Study of Smoothing Methods for Language Models Applied to Information Retrieval. *ACM Transactions on Information Systems*. 2, 2 (2004), 179-214.

[26] Zhai, C. 2008. Statistical Language Models for Information Retrieval A Critical Review. *Found. Trends Inf. Retr.* 2, (Mar. 2008), 137–213.